

## CLAIMS

What is claimed is:

- 1           1. A method, comprising:
  - 2           determining a new enqueue slot of a circular queue having N slots into which a
  - 3           queue element may be enqueued;
  - 4           determining whether the circular queue is full via executing a check comparing
  - 5           relative positions of the new enqueue slot and a current dequeue slot ("CDS"); and
  - 6           enqueueing the queue element into the new enqueue slot, if the circular queue is
  - 7           not full.
- 1           2. The method of claim 1 wherein determining whether the circular queue is full
  - 2           via executing the check comprises determining whether enqueueing the queue element
  - 3           into the new enqueue slot would result in an overflow condition of the circular queue via
  - 4           executing the check.
- 1           3. The method of claim 2, further comprising:
  - 2           setting a last enqueue slot ("LES") pointer currently designating an old enqueue
  - 3           slot to designate the new enqueue slot after determining the new enqueue slot;
  - 4           dropping the enqueue element, if the overflow condition would result from
  - 5           enqueueing the queue element into the new enqueue slot; and
  - 6           resetting the LES pointer to designate the old enqueue slot, if the overflow
  - 7           condition would result from enqueueing the queue element.

1           4. The method of claim 3, wherein executing the check further comprises  
2   determining whether the following relation is true:

$$3 \qquad ((CDS^N - LES^N) \bmod N) < M,$$

4            wherein  $CDS^N$  represents  $CDS \bmod N$ ,  $LES^N$  represents  $LES \bmod N$ , and M  
5            represents a number less than N.

5. The method of claim 4 wherein M is equal to or greater than a maximum number of slots that may be enqueued with queue elements during a delay period for updating a dequeue counter.

6. The method of claim 1 wherein determining the new enqueue slot of the circular queue into which the queue element may be enqueued comprises determining the new enqueue slot according to a pre-sort deficit round robin enqueueing scheme.

7. The method of claim 2 wherein each of the N slots of the circular queue can buffer multiple queue elements corresponding to multiple logical queues, wherein the queue element corresponds to a particular one of the multiple logical queues, and wherein the new enqueue slot corresponds to the particular one of the multiple logical queues.

1           8. The method of claim 7 wherein determining whether the circular queue is full  
2   comprises determining whether enqueueing the queue element into the new enqueue slot

3 of the circular queue would result in an overflow condition of the particular one of the  
4 multiple logical queues.

1 9. A method, comprising:  
2 dequeuing a queue element from a current dequeue slot (“CDS”) of a circular  
3 queue having N slots;  
4 designating a new CDS;  
5 determining whether the circular queue is empty via executing a first check  
6 comparing relative positions of the new CDS and a last enqueued slot (“LES”); and  
7 setting the LES to the new CDS, if the circular queue is determined to be empty.

1 10. The method of claim 9 wherein the CDS is designated by a CDS pointer,  
2 wherein designating the new CDS comprises incrementing the CDS pointer to designate  
3 the new CDS, wherein the LES is designated by a LES pointer, and wherein setting the  
4 LES to the new CDS comprises setting the LES pointer to designate the new CDS, if the  
5 circular queue is determined to be empty.

1 11. The method of claim 9 wherein determining whether the circular queue is  
2 empty further comprises executing a second check prior to executing the first check, the  
3 second check comprising:  
4 determining whether an enqueue count is equal to a dequeue count.

1           12. The method of claim 11, further comprising incrementing the dequeue count  
2 after dequeuing the queue element from the CDS of the circular queue.

1           13. The method of claim 9 wherein executing the first check further comprises  
2 determining whether the following relation is true:

3 
$$((CDS^N - LES^N) \bmod N) < M$$

4           wherein  $CDS^N$  represents  $CDS \bmod N$ ,  $LES^N$  represents  $LES \bmod N$ , and  $M$   
5 represents a number less than  $N$ .

1           14. The method of claim 9 wherein  $M$  is equal to or greater than a maximum  
2 number of slots of the circular queue that may be enqueued with queue elements during a  
3 delay period for updating a dequeue counter.

1           15. The method of claim 9 wherein each of the  $N$  slots of the circular queue can  
2 buffer multiple queue elements corresponding to multiple logical queues, wherein the  
3 queue element corresponds to a particular one of the multiple logical queues, and wherein  
4  $LES$  corresponds to the particular one of the multiple logical queues.

1           16. The method of claim 15 wherein determining whether the circular queue is  
2 empty via executing the first check comprises determining whether the particular one of  
3 the multiple logical queues is empty via executing the first check.

1           17. A machine-accessible medium that provides instructions that, if executed by a  
2 machine, will cause the machine to perform operations comprising:  
3           dequeuing a first queue element from a current dequeue slot (“CDS”) of a circular  
4 queue having N slots, the CDS designated by a CDS pointer;  
5           incrementing the CDS pointer to designate a new CDS; and  
6           determining whether the circular queue is empty after the incrementing via  
7 executing a first check comparing relative positions within the circular queue designated  
8 by the CDS pointer and a last enqueued slot (“LES”) pointer.

1           18. The machine-accessible medium of claim 17, further providing instructions  
2 that, if executed by the machine, will cause the machine to perform further operations,  
3 comprising:  
4           setting the LES pointer currently designating an old enqueue slot of the circular  
5 queue to designate a new enqueue slot of the circular queue into which a second queue  
6 element may be enqueued;  
7           determining whether enqueueing the second queue element into the new enqueue  
8 slot would result in an overflow condition of the circular queue via re-executing the first  
9 check after setting the LES pointer to designate the new enqueue slot.

1           19. The machine-accessible medium of claim 18, further providing instructions  
2 that, if executed by the machine, will cause the machine to perform further operations,  
3 comprising:

4 enqueueing the second queue element into the new enqueue slot, if the overflow  
5 condition would not result from enqueueing the second queue element into the new  
6 enqueue slot;

7 dropping the second enqueue element, if the overflow condition would result from  
8 enqueueing the second queue element into the new enqueue slot; and

9 resetting the LES pointer to designate the old enqueue slot, if the overflow  
10 condition would result from enqueueing the second queue element into the new enqueue  
11 slot.

1 20. The machine-accessible medium of claim 18, further providing instructions  
2 that, if executed by the machine, will cause the machine to perform a further operation,  
3 comprising determining the new enqueue slot of the circular queue into which the second  
4 queue element may be enqueued.

1 21. The machine-accessible medium of claim 20, wherein determining the new  
2 enqueue slot of the circular queue into which the second queue element may be enqueued  
3 comprises determining the new enqueue slot according to a per-sort deficit round robin  
4 queuing scheme.

1 22. The machine-accessible medium of claim 17, further providing instructions  
2 that, if executed by the machine, will cause the machine to perform further operations,  
3 comprising:

```

4         setting the LES pointer to designate the current dequeue slot, if the circular queue
5         is determined to be empty.

```

1           23. The machine-accessible medium of claim 22 wherein the determining  
2   whether the circular queue is empty further comprises executing a second check prior to  
3   executing the first check, the second check comprising:

4 determining whether an enqueue count is equal to a dequeue count, wherein the  
5 enqueue count is incremented each time a queue element is enqueued and the dequeue  
6 count is incremented each time a queue element is dequeued.

1           24. The machine-accessible medium of claim 18 wherein executing and re-  
2   executing the first check comparing the relative positions within the circular queue  
3   designated by the CDS pointer and the LES pointer comprises determining whether the  
4   following relation is true:

$$5 \quad ((CDS^N - LES^N) \bmod N) < M$$

6            wherein  $CDS^N$  represents  $CDS \bmod N$ ,  $LES^N$  represents  $LES \bmod N$ , and  $M$   
7            represents a number less than  $N$ .

1           25. The machine-accessible medium of claim 18 wherein each of the N slots of  
2   the circular queue can buffer multiple queue elements corresponding to multiple logical  
3   queues, and wherein the first queue element, the second queue element, and the LES  
4   pointer correspond to a particular one of the multiple logical queues.

1           26. The machine-accessible medium of claim 25 wherein determining whether  
2           the circular queue is empty comprises determining whether the particular one of the  
3           logical queues is empty and wherein determining whether enqueueing the second queue  
4           element would result in an overflow condition of the circular queue comprises  
5           determining whether enqueueing the second queue element would result in an overflow  
6           condition of the particular one of the logical queues.

1           27. A router system, comprising:  
2           an input port to receive a first data unit from a first network link;  
3           a circular queue having N slots to queue a first queue element;  
4           a network processor communicatively coupled to the input port and the circular  
5           queue, the network processor coupled to:  
6                 set a last enqueue slot ("LES") pointer to designate a new enqueue slot of  
7                 the N slots into which the first queue element may be enqueue; and  
8                 determine whether enqueueing the first queue element into the new  
9                 enqueue slot would result in an overflow condition of the circular queue via  
10                executing a first check after setting the LES pointer, the first check comparing  
11                relative positions within the circular queue designated by the LES pointer and  
12                a current dequeue pointer ("CDS"); and  
13           an output port communicatively coupled to the network processor to transmit the  
14           first data unit to a second network link in response to dequeuing the first queue element  
15           from the circular queue.

1           28. The router system of claim 27 wherein the network processor is further  
2 coupled to:  
3           enqueue the first queue element into the new enqueue slot, if the overflow  
4 condition would not result from the enqueueing the first queue element into the new  
5 enqueue slot;  
6           drop the first data unit, if the overflow condition would result from enqueueing the  
7 first queue element into the new enqueue slot; and  
8           reset the LES pointer to designate a previous enqueue slot, if the overflow  
9 condition would result from enqueueing the first queue element into the new enqueue slot.

1           29. The router system of claim 28 wherein the network processor is further  
2 coupled to:  
3           dequeue a second queue element from a CDS of the N slots designated by the  
4 CDS pointer;  
5           increment the CDS pointer to designate a new CDS; and  
6           determine whether the circular queue is empty via re-executing the first check  
7 after the increment of the CDS pointer.

1           30. The router system of claim 28 wherein the network processor is further to  
2 determine whether the circular queue is empty via executing a second check prior to re-  
3 executing the first check, the second check comprising:

4 determining whether an enqueue count is equal to a dequeue count, wherein the  
5 network processor is to increment the enqueue count each time a queue element is  
6 enqueued and to increment the dequeue count each time a queue element is dequeued.

1 31. The router system of claim 27 wherein executing the first check comparing  
2 the relative positions within the circular queue designated by the CDS pointer and the  
3 LES pointer comprises determining whether the following relation is true:

$$4 \quad ((CDS^N - LES^N) \bmod N) < M$$

5 wherein  $CDS^N$  represents  $CDS \bmod N$ ,  $LES^N$  represents  $LES \bmod N$ , and  $M$   
6 represents a number less than  $N$ .

1 32. The router system of claim 27 wherein the first data unit comprises one of a  
2 packet, a cell, and a frame.

1 33. The router system of claim 27 wherein the first queue element comprise a  
2 pointer to a memory location containing the data unit.

1 34. The router system of claim 29 wherein each of the  $N$  slots of the circular  
2 queue can buffer multiple queue elements corresponding to multiple logical queues and  
3 wherein the first queue element, the second queue element, and the LES pointer  
4 correspond to a particular one of the multiple logical queues.

1           35. The router system of claim 34 wherein determining whether the circular  
2 queue is empty comprises determining whether the particular one of the logical queues is  
3 empty and wherein determining whether enqueueing the first queue element would result  
4 in an overflow condition of the circular queue comprises determining whether enqueueing  
5 the first queue element would result in an overflow condition of the particular one of the  
6 logical queues.